Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ФУНДАМЕНТАЛЬНЫЕ НАУКИ

КАФЕДРА МАТЕМАТИКА И КОМПЬЮТЕРНЫЕ НАУКИ

Отчет

по лабораторному заданию № 13

Вариант 18

Дисциплина: Информатика

Название лабораторного задания: Использование указателей при работе со строками C++. Динамические строки. Текстовая обработка.

Студент гр. ФН11-22Б — 10.03.21 — **М.Х. Хаписов**
(Подпись, дата) (И.О. Фамилия)

Преподаватель Доцент кафедры ИУ-6 — 10.03.21 — **Т.Н. Ничушкина**
(Подпись, дата) (И.О. Фамилия)

Москва, 2021

Цель работы:

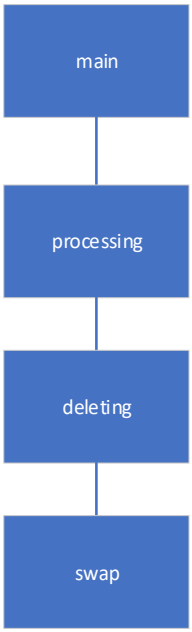
- Приобретение навыков работы с динамической памятью и указателями на языке программирования C++.
- Изучение принципов и приемов работы с динамическими строками, а также изучение способов обработки строк.

Задание: С клавиатуры вводится текст (последовательность строк, содержащих буквы латинского алфавита и символ «отмена предыдущего символа» - #). Напечатать строки, удалив указанные символы:

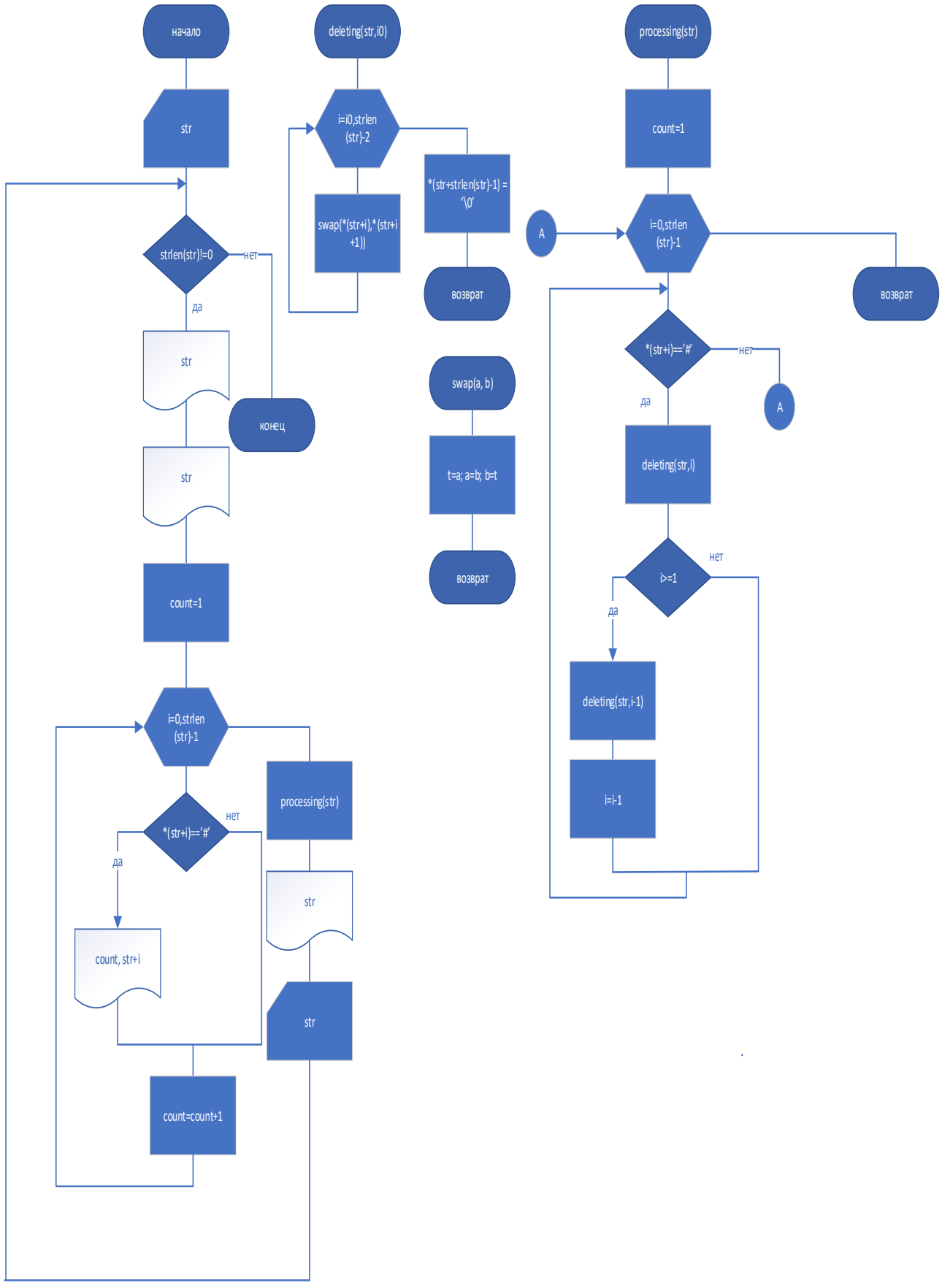
На#ejj##llo => Hello.

Пользуясь **указателями**, вывести на экран адреса найденных символов #.

Структурная схема



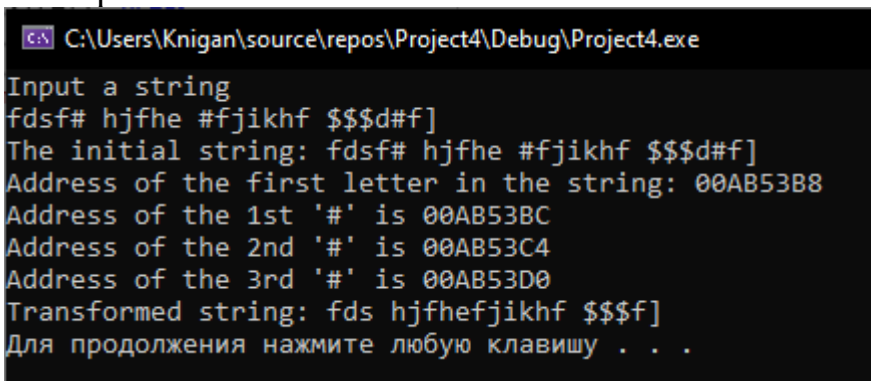
Схемы алгоритмов


начало
str
strlen(str)!=0
нет
да
str
конец
str
count=1
i=0,strlen (str)-1
*(str+i)==’#’
нет
да
count, str+i
count=count+1
processing(str)
str
str
deleting(str,i0)
i=i0,strlen (str)-2
swap(*(str+i),*(str+i +1))
*(str+strlen(str)-1) = ‘\0’
возврат
swap(a, b)
t=a; a=b; b=t
возврат
processing(str)
count=1
A
i=0,strlen (str)-1
возврат
*(str+i)==’#’
нет
да
A
deleting(str,i)
i>=1
нет
да
deleting(str,i-1)
i=i-1

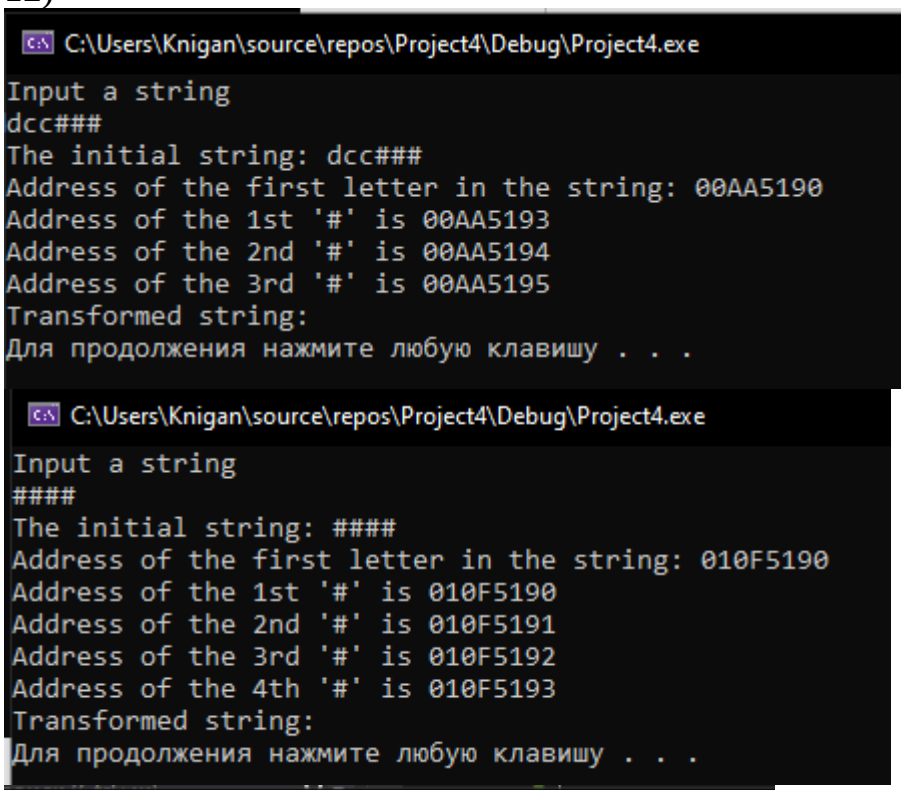
Текст программы

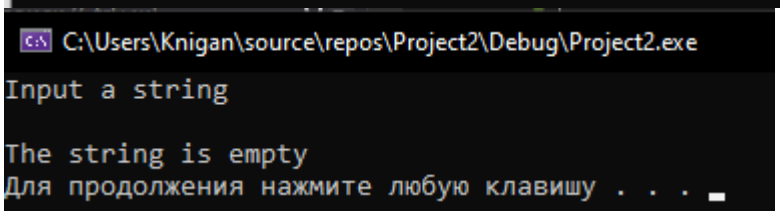
```
#include <iostream>
void swap(char &a,char &b) {
    char t=a;
    a=b; b=t;
}
void deleting(char* str,size_t i0) {
    for(size_t i=i0;i<strlen(str)-1;i++) swap(*(str+i),*(str+i+1));
    *(str+strlen(str)-1)='\0';
}
void processing(char *str) {
    for(size_t i=0;i<strlen(str);i++) {
        while(*(str+i)=='#') {
            deleting(str,i);
            if(i>=1) {
                deleting(str,i-1);
                i--;
            }
        }
    }
}
void main() {
    char *st=new char[1024];
    puts("Input a string");
    gets_s(st,1024);
    while (strlen(st)!=0) {
        char *str=new char[strlen(st)];
        for (size_t i=0;i<strlen(st);i++) *(str+i)=*(st+i);
        *(str+strlen(st))='\0';
        printf("The initial string: %s\n",str);
        printf("Address of the first letter in the string: %p\n",str);
        size_t count=1;
        for (size_t i=0;i<strlen(str);i++) {
            if (*(str+i)=='#') {
                switch(count) {
                    case 1: printf("Address of the 1st '#' is %p\n",str+i); break;
                    case 2: printf("Address of the 2nd '#' is %p\n",str+i); break;
                    case 3: printf("Address of the 3rd '#' is %p\n",str+i);  break;
                    default: printf("Address of the %dth '#' is %p\n",count,str+i); break;
                }
                count++;
            }
        }
        processing(str);
        printf("Transformed string: %s\n",str);
        puts("Input a string");
        gets_s(st,1024);
    }
    system("pause");
}
```

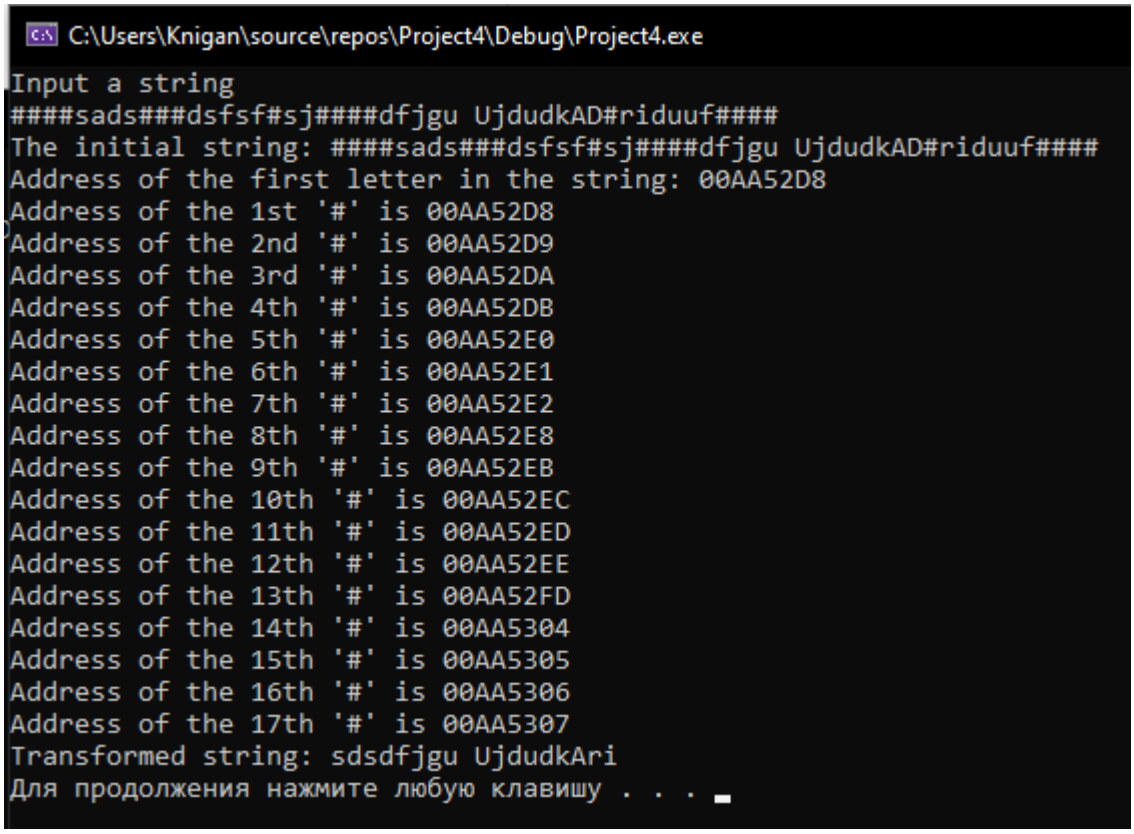
Тестирование



Разница адресов первой и второй решётки составляет |00AB53C4 – 00AB53BC| = 8 (что соответствует её положению в строке), аналогично соответствует и третья решётка (разница между второй и третьей составляет |00AB53D0 – 00AB53C4| = 12)







Символы в строке располагаются в памяти последовательно, что легко видеть на примере символа ‘#’. Подряд идущие в строке 4 # (первые 4 символа) в памяти располагаются также последовательно, а адрес пятой решётки больше на |00AA52E0 – 00AA52DA| = 5 адреса четвёртой ячейки, как и должно быть при последовательном расположении символов в памяти.

Вывод: я приобрёл навыки работы с динамической памятью и указателями на языке программирования C++, изучил принципы и приёмы работы с динамическими строками, а также изучил способы обработки строк

Ответы на контрольные вопросы

1. Каковы особенности реализации «строки» в C++? (Ответ: Символьная строка в C++ определяется как массив символов, который заканчивается нуль-символом, поэтому размер символьного массива всегда должен быть на один символ больше, чем требуется для хранения строки)

2. Что такое нуль-символ и зачем он используется? (Нуль-символ – это маркер конца строки «\0». Если маркер конца строки в символьном массиве отсутствует, то такой массив строкой не является)
3. Какой синтаксис имеет описание строк?

char *<имя указателя> =new char[(<Количество>)];

4. Чем различаются статические и динамические строки? (Ответ: статическая строка имеет заранее определённый размер, когда размер динамической строки может задаваться пользователем, и, более того, размер такой строки можно как увеличивать, так и уменьшать во время работы программы)
5. Какие функции используются для ввода и вывода строк? Какую библиотеку необходимо для этого подключить? (Ответ: функции gets_s и puts соответственно при помощи библиотеки stdio.h)